REMARKS/ARGUMENTS

On page 2 of the Official Action, claim 6 was rejected under 35 U.S.C. 112, second paragraph, as being indefinite. The Official Action says the phrase "grouping the on-demand anti-virus scan request into chunks of multiple ones of the on-demand anti-virus scan requests" can be interpreted as grouping the individual scan request or grouping using chunks of multiple ones. In reply, claims 6, 8, 12, 22, and 24 have been amended to recite "grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, ..." It is respectfully submitted that the plain ordinary meaning of the verb "to group" is "combine in a group" where the meaning of the noun "group" is "an assemblage of objects regarded as a unit." For example, it is often said that players are grouped into teams. Thus, applicant's claim language should be understood as calling for grouping of the on-demand anti-virus scan requests. The groups of the on-demand anti-virus scan requests are called chunks. Support for this amendment is found in applicant's original specification on page 13, lines 13-22, as follows:

> An on-demand request chunking routine 67 services the on-demand request queue 66 by grouping on-demand anti-virus file scan requests into chunks, and when the number of anti-virus file scan requests in the virus scan request queue is less than a threshold, by inserting each chunk of on-demand anti-virus file scan requests into the virus scan request queue 63.
>
> **[00034]** For example, there are ten anti-virus threads in the anti-virus thread pool 64, there are five virus checkers, and the chunk size is fifty anti-virus file scan requests. The threshold is set to twenty-five. In a more general case of

> "N" virus checkers and "M" anti-virus threads, for example, the chunk size is the product (M*N), and the threshold is set to one-half of the chunk size.

Claim 7 has been amended to depend on claim 6 for proper antecedent basis for chunks. Support for this amendment is also found in original claims 22 and 23 and in Fig. 8 step 122 and paragraph [00047] on page 17 lines 12-20 of applicant's specification.

On page 3 of the Official Action, claims 1-28 were rejected under 35 U.S.C. 103(a) as being unpatentable over Edwards (U.S. Pat. 7,188,367) in view of Novell , "Product Focus: Antivirus Solutions for NetWare," 01 Jan. 1999. Applicant respectfully traverses.

The policy of the Patent and Trademark Office has been to follow in each and every case the standard of patentability enunciated by the Supreme Court in Graham v. John Deere Co., 148 U.S.P.Q. 459 (1966). M.P.E.P. § 2141. As stated by the Supreme Court:

> Under § 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.

148 U.S.P.Q. at 467.

Edwards discloses a virus scanner in which a pool of pre-processor threads and a queue are interposed between the event filter and the pool of scanner threads. The pre-processor threads perform operations that can be completed quickly to determine whether an object of a scan

request needs to be scanned. The pre-processor threads gather characteristics about the scan requests and place them in the queue in a priority order based on those characteristics. The scanner threads select a scan request from the queue based on the priority order. Alternatively, the scan request is selected based on the scan request's characteristics as compared to the characteristics of the scan requests whose objects are currently being scanned by other scanner threads in the pool. (Edwards, Abstract.)

Novell says: "Command AntiVirus with F-PROT Professional 4.52 for NetWare from Command Software Systems Inc. is an enterprise-wide antivirus solution. Command AntiVirus with F-PROT Professional 4.52 for NetWare offers a heuristic analysis, which includes both an on-access scanner and an on-demand scanner to search for polymorphic viruses. In addition, you can configure multiple concurrent scans."

Applicant's claim 1 defines a method of operating a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning. The method comprises combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue; and distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers.

Regarding the differences between the cited art and the subject matter of applicant's claim 1, the Official Action says that although Edwards does teach both on-demand and on-access virus scan requests, Edwards does not explicitly teach wherein the scan request queue has on-demand anti-virus scanning concurrent with on-access anti-virus scanning. Edwards, Col. 1, lines 43-48, say: "Virus scanners may be invoked on-demand by a computer user to scan a selected file. More typically, virus scanners install themselves as part of an operating system and

then scan files, according to user preferences, as the files are created and accessed. This type of virus scanner is referred to as an on-access virus scanner."

Applicant respectfully submits that the system shown in FIG. 1 of Edwards and labeled "Prior Art" is said to be an on-access virus scanner. Edwards, col. 2, lines 12-24, say:

> A prior art on-access virus scanner 100 is organized into two parts, as illustrated in FIG. 1. One part is the event filter 110, which is the software that intercepts the events of interest to the virus scanner. Events of interest include a file being opened or an e-mail arriving in a mailbox. Another part is a scanner thread 120, which is the software that receives scan requests from the filter. The scanner thread determines whether the object of the intercepted event (i.e. the file, e-mail, or e-mail attachment) needs scanning and, if so, scans the object. Multiple scanner threads are typically provided in pools 130 that are capable of executing concurrently so that multiple objects may be scanned simultaneously.

As further disclosed in Edwards column 2, lines 25-37, Edwards is directed to a problem that arises in the on-access virus scanner:

> Unfortunately, virus developers have recently begun to manufacture "malicious" files which take "a long time" to scan, including archives and documents containing embedded objects. The malicious files are designed to overwhelm on-access virus scanners by tying up all of the available scanner threads in the pool, thereby causing all other events intercepted by the filter to be queued until a scanner thread becomes free. This causes the virus scanner to "crash" by blocking further processing of data and leaves a system undefended against subsequent attacks. If e-mail or file processing is routed through a virus

scanner and the scanner has crashed, then a "denial of service" for e-mail or file activity occurs until the scanner is restarted.

Edward's embodiment disclosed in FIGS. 2-6 also appears to be directed to an on-access virus scanner because the disclosed embodiment includes an event filter (Edwards, FIG. 2, box 110). Edwards col. 4, lines 50-53 say: "A scan request is generated whenever an event occurs that causes the host system to access an object, such as opening a file, reading an e-mail, or opening an e-mail attachment." Edwards col. 4 lines 64-67 say: "As illustrated, a pre-processor pool 230 of four pre-processor threads 210 and a priority queue 220 are interposed between the event filter 110 and the scanner thread pool 130 of three scanner threads 120." Edwards col. 10, lines 3-7 say: "For example, while the foregoing description focused on on-access virus scanners, it will be recognized that the above techniques and analyses can be applied to scanning data in other contexts such as on-demand virus scanners having comparable limitations."

Novell also teaches that on-demand scanners are different from on-access scanners. Although Novell says you can configure multiple concurrent scans, there is no suggestion that the on-access scanner would ever concurrently scan for an on-demand anti-virus scan request, nor is there any suggestion that the on-demand scanner would ever concurrently scan for an on-access anti-virus scan request.

Page 4 of the Official Action says it would have been obvious to one of ordinary skill in the art to combine the virus scan queue of Edwards to include concurrent on-access and on-demand scanners. Applicant respectfully disagrees. Edwards teaches that a particular problem with the Edwards FIG. 1 prior art on-access anti-virus scanner can be solved by adding a priority

queue 220 to produce the on-access antivirus scanner shown in Edwards FIG. 2. Although it would be possible for an on-demand anti-virus scanner to have a queue, it is not understood how the virus scan queue of Edwards would or should be modified to include an on-access anti-virus scanner and an on-demand anti-virus scanner. More importantly, applicant's claim 1 is not calling for a virus scan queue including concurrent on-access and on-demand scanners. Instead, applicant's claim 1 calls for combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue, and distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers (for on-demand anti-virus scanning concurrent with on-access anti-virus scanning).

Page 4 of the Official Action says: "The motivation to combine is that it is well known in the virus scanning art to have both on-demand and on-access scanners running concurrently. Novell provides an example of such a system." Although it should be possible to run the on-demand scanner of Novell concurrently with the on-access scanner of Novell in a multi-processing computer system, this would not have motivated one of ordinary skill to combine on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue, and distribute the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the Novell on-demand checker and the Novell on-access checker running concurrently. From the teaching in Novell, one would simply send on-demand anti-virus scan requests to the Novell on-demand scanner, and the Novell on-access scanner would scan in response to any on-access events. If the Novell on-access scanner would have a problem as described in Edwards, then the Novell on-access scanner could be modified as described in Edwards, but that would not result in combining on-demand anti-virus scan requests

and on-access anti-virus scan requests in a virus scan request queue, as called for in applicant's claim 1.

"[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." In re Kahn, 441 F. 3d 977, 988 (Fed. Cir. 2006). A fact finder should be aware of the distortion caused by hindsight bias and must be cautious of arguments reliant upon ex post reasoning. See KSR v. Teleflex, 550 U.S. __ (2007), citing Graham, 383 U. S. at 36 (warning against a "temptation to read into the prior art the teachings of the invention in issue" and instructing courts to "guard against slipping into the use of hindsight.").

In contrast to Edwards and Novel, the applicant's specification teaches that it is desirable to use the same virus checkers for on-demand and on-access virus checking. This leads to several problems because the priority and workload for the on-demand scan requests are so much different from the priority and workload for the on-access scan requests. The different priority and workload would suggest that the on-demand scan requests should be treated differently from the on-access scan requests. Nevertheless, the applicant's specification teaches that it is desirable to combine the on-demand requests and the on-access requests in a queue under appropriate conditions. This is explained in applicant's specification on page 11 line 20 to page 12 line 23 as follows:

> **[00030]** The scanning task shown in FIG. 2 is generally referred to as an "on-access" virus scan. An "on-access" virus scan is processed in real-time when scanning is triggered by user-initiated file access. Another kind of virus scan is known as an "on-demand" virus scan. An "on-demand" virus scan is scheduled at a lower priority than "on-access," and it typically involves scanning all files of

virus-checkable file type in a one or more specified file systems. For example, "on-demand" virus checking is scheduled when a new virus is discovered, when new unchecked files are migrated into a file server, or prior to archiving or backing-up unchecked files. Although lower in priority from a scheduling point of view, "on-demand" virus checking has often been more burdensome on the data processing system than "on-access" virus checking. A full file system scan may generate a much more intense scanning load when a multitude of files in the file system must be scanned. Even though this scanning workload is distributed over multiple virus checkers, the volume of scans will generate a significant resource load on the operating system of the data mover. Moreover, it is desirable to fully utilize the capabilities of the virus checkers in order to complete the full file system scan as soon as possible

[00031] In order to mitigate any general performance degradation on the data mover during user file access, it is desirable to mix "on-demand" virus scan requests with "on-access" virus scan requests in a shared virus scan request queue. For example, outstanding "on-demand" virus scan requests are added to the shared queue when the number of requests in the shared queue falls below a threshold. The threshold is selected to provide a relatively continuous flow of requests to the virus checkers without significantly degrading the response time of the virus checkers for responding to the "on-access" requests. Moreover, it is desirable to add outstanding "on-demand" virus scan requests to the shared queue in manageable "chunks", and to wait until the virus scan requests in each chunk have been serviced before sending another chunk of "on-demand" virus scan requests.

Neither Edwards nor Novell suggest these problems nor the particular solution as defined in applicant's claim 1. The fact that the on-demand scan requests have priority and workloads that are so much different from the priority and workloads of the on-access scan requests, and the fact

that Novell teaches that an on-demand virus checker should be used for on-demand scan requests and an on-access virus checker should be used for on-access scan requests, teach away from treating the on-demand scan requests and the on-access scan requests in a similar fashion by combining them in a virus scan request queue for distribution to virus checkers. Edwards says nothing contrary to the teaching of Novel, which is about four years prior to the filing of the applicant's patent application. In short, the teachings of Edwards and Novell and the nature of the problem solved by the applicant show that the subject matter of the applicant's claim 1 would not have been obvious.

With reference to applicant's claim 5, it is not seen where Edwards or Novell gives on-access anti-virus scan requests priority over on-demand anti-virus scan requests. Nor is it understood how Edwards col. 5, lines 66-67 or col. 6, lines 1-3 discloses inhibiting the placement of any particular kind of virus scan request onto the virus scan request queue when the number of anti-virus scan requests on the queue reaches a threshold. Edwards col. 5, line 65 to col. 6, line 3 say: "For example, using the scan request's user characteristics, a pending scan request from user A may be determined to be more suitable than a pending scan request from user B if three of the four scanner threads are already scanning scan requests from user B. This prevents a single user B from monopolizing the virus scanner 200."

Other passages of Edwards disclose that event filter 110 or scan prioritizer 240 places the on-access virus scan requests on the virus scan request queue, and some of the virus scan requests on the queue are given priority over other virus scan requests on the priority queue because the event filter or scan prioritizer places the virus scan requests on the queue in the order

that the threads should process them, and because the scanner threads select for scanning particular virus scan requests that are on the queue based on the position of the virus scan requests on the queue or the operational characteristics of the virus scan requests. For example, Edwards col. 5, lines 39-43 say: "In one embodiment, the characteristics obtained by the pre-processor threads 210 are used by the event filter 110 to prioritize the scan requests by placing them in the priority queue 220 in the order in which the scanner threads 120 should process them." Edwards col. 6 lines 23-29 say: "As another example, using the scan request's operational characteristics, such as a time stamp of when the scan request was triggered by the event filter 110 or when it was placed on the priority queue 220, scan requests that have been passed over too often (i.e. that have been on the priority queue 220 the longest) could eventually be given higher priority than scan requests that would otherwise come first." Edwards col. 6 lines 42-46 say: "In each of the above examples, the scanner threads 120 process scan requests either by selecting the most suitable pending request, or by selecting the next pending request that was placed on the priority queue in the optimal priority order by the scan prioritizer 240."

With reference to applicant's claim 6, Edwards fails to discloses grouping on-demand anti-virus scan requests into chunks, and placing the chunks onto the virus scan request queue. For the reasons discussed above with respect to applicant's claim 1, placing on-demand anti-virus scan requests on Edward's priority queue of on-access anti-virus scan request is not suggested by the fact that Edwards places or selects for scanning on-access anti-virus requests on the queue based on operational characteristics of the on-access anti-virus scan requests. Nor is it seen where Edwards discloses grouping any particular kind of request into chunks, and then

placing such chunks onto the virus scan request queue. Instead, as discussed above, Edwards discloses that the event filter or scan prioritizer places each on-access virus scan request on the priority queue, and a scanner thread selects a virus scan request on the priority queue to process based on the position of the virus scan request on the queue and on the scan request's operational characteristics. In particular, the embodiment of Edwards in which the scan prioritizer places the scan request on the priority queue in some pre-defined fixed order other than the order in which the scan request was received, e.g. executable files first and data files second (Edwards, col. 7, lines 31-37), is different from grouping the data files into a chunk, and placing the chunk on the queue. Nor is grouping the data files into a chunk, and placing the chunk on the queue "inherent," because the method disclosed in Edwards in fact obtains executable files first and data files second in the queue without grouping the data files into a chunk, and placing the chunk on the queue. In accordance with the method disclosed in Edwards, when the scan prioritizer is placing a new request to scan an executable file in the queue and the queue already contains one or more requests for scanning data files, the scan prioritizer places the request to scan the executable file in the queue at a position before the one or more requests for scanning data files.

Applicant's claim 7 calls for inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks. This is clearly different from simply scanning one type of grouped scans before scanning another type of grouped scans. Moreover, applicant's chunks of claim 7 as amended are of the same type, namely, chunks of on-demand virus scan requests.

Applicant's claim 8 calls for distributing on-demand anti-virus scan requests and on-access anti-virus scan requests to virus checkers so that the virus checkers perform on-demand anti-virus scanning concurrent with on-access anti-virus scanning. Claim 8 also calls for distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers. Thus, claim 8 calls for at least one virus checker that performs the concurrent on-access virus scanning to also perform an on-demand anti-virus scan, because the multiple ones of the on-demand anti-virus scan requests are distributed over the virus checkers. As discussed above with respect to applicant's claim 1, neither Edwards nor Novell suggests that an on-access virus checker should perform on-demand antivirus checking.

Applicant's claim 8 as amended further calls for "grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and for each chunk, distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers." As discussed above with respect to applicant's claim 6, neither Edwards nor Novell suggests grouping of on-demand virus scan requests into chunks, and for each chunk, distributing the on-demand anti-virus scan requests over the virus checkers. In particular, Edwards giving priority to one type of on-access scans (scans of executable files) over another type of on-access scans (scans of data files) to solve a particular problem in an on-access virus checker does not suggest that on-demand virus scans should be grouped into chunks for distribution of the on-demand virus scan requests for each chunk over virus checkers that concurrently perform on-demand and on-access virus checking.

With respect to applicant's claim 11, see applicant's remarks above with respect to applicant's claim 7.

With respect to applicant's claim 12, see applicant's remarks above with respect to applicant's claims 1, 5 and 6.

With respect to applicant's claim 15, see applicant's remarks above with respect to applicant's claim 7.

With respect to applicant's claim 16, see applicant's remarks above with respect to applicant's claim 1.

With respect to applicant's claim 17, the virus checkers of Edwards or Novell would not be separate from the file server containing said at least one processor and said virus scan request queue if the file server also contained the virus checkers. The fact that Edwards has pre-processor threads that obtain certain additional characteristics about the scan requests, such as whether the object is being accessed from the server console or from a network client, does not suggest that that the processor and the virus scan request queue are in a file server and the virus checkers are separate from the file server. Instead, Edwards FIG. 2 shows that the virus scanners are scanner threads which together with the priority queue and pre-processor threads and event filter/prioritizer and scan prioritizer comprise an on-access virus scanner 200, suggesting that the processor that services the priority queue also executes the pre-processor threads and the scanner

threads. See also Edwards col. 7 lines 6-10 and 60-63, col. 8 lines 22-54, and col. 9 lines 35-40 ("Computer system 600 and virus scanning software 200/300/400/500 stored and executed therein as part of the method and apparatus of the present invention …").

With respect to applicant's claim 21, see applicant's remarks above with respect to applicant's claim 5.

With respect to applicant's claim 22, see applicant's remarks above with respect to applicant's claim 6.

With respect to applicant's claim 23, see applicant's remarks above with respect to applicant's claim 7.

With respect to applicant's claim 24, see applicant's remarks above with respect to applicant's claims 1, 6, and 17.

With respect to applicant's claim 27, see applicant's remarks above with respect to applicant's claim 5.

With respect to applicant's claim 28, see applicant's remarks above with respect to applicant's claim 7.

In view of the above, it is respectfully submitted that the application is in condition for allowance. Reconsideration and early allowance are earnestly solicited.

Respectfully submitted,

/ Richard C. Auchterlonie /

Richard C. Auchterlonie, Reg. No. 30,607

NOVAK DRUCE & QUIGG, LLP
1000 Louisiana, 53rd Floor
Houston, TX 77002
713-571-3460